AMENDMENTS TO THE CLAIMS

1. (Currently Amended) A method comprising:

storing native code associated with a first method and a second method within a native code space;

~~creating~~ associating a ~~symbolic~~ reference to the first method and the second method in a method table;

determining whether the first method or the second method is invoked by detecting whether the first method or the second method correspond to the native code;

determining whether the native code space exceeds a threshold in response to an invocation of ~~a~~ the first method or the second method;

incrementing method counters each time the first method or the second method is invoked, wherein the method counters correspond to the first method and the second method;

unwinding a stack to determine whether the first method or the second method is active based on whether a corresponding method counter has exceeded a count threshold;

reclaiming the native code associated with the first method and compiling byte code into native code associated with the second method in response to determining that the second method is active; and

updating the method table for the first method ~~to reference an appropriate symbolic reference~~.

2. (Previously Presented) The method of claim 2, wherein reclaiming the native code associated with the first method and compiling byte code into native code associated with the second method in response to the determination comprises reclaiming the native code associated with the first method in response to a determination that the native code space exceeds the threshold.

3. (Previously Presented) The method of claim 2, further comprising storing the native code associated with the second method within the native code space in response to the compilation.

4. (Previously Presented) The method of claim 2, further comprising:
invoking the first method following the reclamation; and
re-compiling byte code into the native code associated with the first method in response to the invocation of the first method.

5. (Previously Presented) The method of claim 2, wherein reclaiming the native code associated with the first method and compiling byte code into native code associated with the second method in response to the determination comprises compiling byte code into native code associated with the second method.

6. (Previously Presented) The method of claim 5, wherein compiling byte code into native code associated with the second method comprises compiling byte code into native code associated with the second method utilizing a just-in-time compiler.

7. (Previously Presented) The method of claim 2, wherein reclaiming the native code associated with the first method and compiling byte code into native code associated with the second method in response to the determination comprises:

determining whether the first method is active or inactive; and

reclaiming the native code associated with the first method in response to a determination that the first method is inactive.

8. (Previously Presented) The method of claim 7, further comprising:

reclaiming the native code associated with the first method and compiling byte code into native code associated with the second method in response to the determination further comprises determining whether the first method is hot or cold in response to a determination that the first method is inactive; and

reclaiming the native code associated with the first method in response to a determination that the first method is inactive comprises reclaiming the native code associated with the first method in response to a determination that the first method is cold.

9. (Currently Amended) A machine-readable medium having stored thereon data representing sets of instructions which, when executed by a machine, cause the machine to:

store native code associated with a first method and a second method within a native code space;

~~create~~ associate a ~~symbolic~~ reference to the first method and the second method in a method table;

determining whether the first method or the second method is invoked by detecting whether the first method or the second method correspond to the native code;

determine whether the native code space exceeds a threshold in response to an invocation of ~~a~~ the first method and the second method;

increment method counters each time the first method or the second method is invoked, wherein the method counters correspond to the first method and the second method;

unwind a stack to determine whether the first method or the second method is active based on whether a corresponding method counter has exceeded a count threshold;

reclaim the native code associated with the first method and compiling byte code into native code associated with the second method in response to determining that the second method is active; and

update the method table for the first method ~~to reference an appropriate symbolic reference~~.

10. (Previously Presented) The machine-readable medium of claim 9, wherein reclaiming the native code associated with the first method and compiling byte code into native code associated with the second method in response to the determination comprises reclaiming the native code associated with the first

method in response to a determination that the native code space exceeds the threshold.

11. (Previously Presented) The machine-readable medium of claim 9, wherein the sets of instructions when further executed, cause the machine to perform operations comprising storing the native code associated with the second method within the native code space in response to the compilation.

12. (Previously Presented) The machine-readable medium of claim 9, wherein the sets of instructions when further executed, cause the machine to perform operations comprising invoking the first method following the reclamation; and re-compile byte code into the native code associated with the first method in response to the invocation of the first method.

13. (Previously Presented) The machine-readable medium of claim 9, wherein reclaiming the native code associated with the first method and compiling byte code into native code associated with the second method in response to the determination further cause the machine to compile byte code into native code associated with the second method.

14. (Previously Presented) The machine-readable medium of claim 13, wherein compiling byte code into native code associated with the second method further cause the machine to compile byte code into native code associated with the second method utilizing a just-in-time compiler.

15. (Previously Presented) The machine-readable medium of claim 9, wherein reclaiming the native code associated with the first method and compiling byte code into native code associated with the second method in response to the determination further cause the machine to:

determine whether the first method is active or inactive; and

reclaim the native code associated with the first method in response to a determination that the first method is inactive.

16. (Cancelled)

17. (Currently Amended) A data processing system comprising:

a storage device; and

a processor coupled with the storage device, the processor to process data and execute instructions, the processor to:

store native code associated with a first method and a second method within a native code space ~~of the memory~~;

~~create~~ associate a ~~symbolic~~ reference to the first method and the second method in a method table;

determine whether the first method or the second method is invoked by detecting whether the first method or the second method correspond to the native code;

determine whether the native code space exceeds a threshold in response to an invocation of ~~a~~ the first method and the second method;

~~incrementing~~ increment method counters each time the first method or the second method is invoked, wherein the method counters correspond to the first method and the second method;

~~unwinding~~ unwinding a stack to determine whether the first method or the second method is active based on whether a corresponding method counter has exceeded a count threshold;

~~reclaiming~~ reclaim the native code associated with the first method and compiling byte code into native code associated with the second method in response to determining that the second method is active; and

~~updating~~ update the method table for the first method ~~to reference an appropriate symbolic reference~~.

18. (Previously Presented) The data processing system of claim 17, wherein when reclaiming the native code associated with the first method and compiling byte code into native code associated with the second method in response to the determination, the processor is further to reclaim the native code associated with the first method in response to a determination that the native code space exceeds the threshold.

19. (Previously Presented) The data processing system of claim 17, wherein the processor is further to store the native code associated with the second method within the native code space in response to the compilation.

20. (Previously Presented) The data processing system of claim 17, wherein the processor is further to invoke the first method following the reclamation; and re-compiling byte code into the native code associated with the first method in response to the invocation of the first method.

21. (Previously Presented) The data processing system of claim 17, wherein when reclaiming the native code associated with the first method and compiling byte code into native code associated with the second method in response to the determination, the processor is further to compile byte code into native code associated with the second method.

22. (Previously Presented) The data processing system of claim 21, wherein when compiling byte code into native code associated with the second method, the processor is further to compile byte code into native code associated with the second method utilizing a just-in-time compiler.

23. (Previously Presented) The data processing system of claim 17, wherein when reclaiming the native code associated with the first method and compiling byte code into native code associated with the second method in response to the determination, the processor is further to:

determine whether the first method is active or inactive; and

reclaim the native code associated with the first method in response to a determination that the first method is inactive.

24. (Previously Presented) The data processing system of claim 23, wherein the processor is further to:

reclaim the native code associated with the first method and compiling byte code into native code associated with the second method in response to the determination further comprises determining whether the first method is hot or cold; and

reclaim the native code associated with the first method in response to a determination that the first method is inactive comprises reclaiming the native code associated with the first method in response to a determination that the first method is cold.

Claims 25-28 (Cancelled)